IN THE CLAIMS

Please rewrite the claims as follows:

1. (Currently Amended) A method of performing a peer-to-peer DMA to efficiently move data between a first DMA capable ASIC block and a second DMA capable ASIC block over an AHB bus architecture, said method comprising:

initiating a DMA by setting up a first ASIC block to access a second ASIC block;

holding off the DMA transfer until the second ASIC block is set up and ready for the transfer; and

transferring data between the second ASIC block and the first ASIC block after the second ASIC block is set up and ready for the DMA transfer without accessing system memory.

2. (Original) A method according to claim 1, wherein the first ASIC block initiates the DMA by accessing a specific address that corresponds to the second ASIC block.

3. (Currently Amended) A method ~~according to claim 1~~ of performing a peer-to-peer DMA to efficiently move data between a first DMA capable ASIC block and a second DMA capable ASIC block over an AHB bus architecture, said method comprising:

initiating a DMA by setting up a first ASIC block to access a second ASIC block;

holding off the DMA transfer until the second ASIC block is set up and ready for the transfer; and

transferring data between the second ASIC block and the first ASIC block after the second ASIC block is set up and ready for the DMA transfer without accessing system memory,

wherein a master interface of the first ASIC block is used to initiate the DMA transfer and to generate control signals and data signals for the second ASIC block, and wherein the control signals and data for the second ASIC block are routed through a slave interface of the second ASIC block.

4. (Original) A method according to claim 3, wherein signals of the second ASIC block are mapped from a master interface of the second ASIC block to the slave interface of the second ASIC block through muxes.

5. (Original) A method according to claim 1, wherein the second ASIC block is set up as if it is going to perform a DMA in the opposite direction as the first ASIC block.

6. (Original) A method according to claim 1, wherein holding off the DMA transfer comprises sending a split response until the second ASIC block is set up and ready to perform the DMA transfer.

7. (Currently Amended) A method according to claim ~~1~~ 3, wherein the DMA transfer takes place without accessing system memory.

8. (Original) A method according to claim 1, wherein the DMA transfer is performed without the use of sideband signals.

9. (Original) A method according to claim 1, wherein the DMA transfer is performed without duplicate hardware for interfacing core blocks on another side of the ASIC blocks.

10. (Canceled)

11. (Canceled)

12. (Canceled)

13. (Canceled)

14. (Canceled)

15. (Canceled)

16. (Canceled)

17. (Canceled)

18. (Canceled)

19. (Canceled)

20. (Canceled)

21. (Currently Amended) A method of establishing a peer-to-peer DMA, comprising:

initiating a peer-to-peer DMA from a first DMA capable ASIC block;

setting up a second DMA capable ASIC block to communicate with the initiating ASIC block; and

routing control and data signals between the second ASIC block and the first ASIC block without accessing system memory.

22. (Currently Amended) A method ~~according to claim 21~~ of establishing a peer-to-peer DMA, comprising:

initiating a peer-to-peer DMA from a first DMA capable ASIC block;

setting up a second DMA capable ASIC block to communicate with the initiating ASIC block; and

routing control and data signals between the second ASIC block and the first ASIC block without accessing system memory,

wherein setting up the second DMA capable ASIC block to communicate with the initiating ASIC block comprises configuring the second ASIC block as if it is going to perform a normal DMA in an opposite direction as the initiating ASIC block.

23. (Original) A method according to claim 21, wherein routing the control and data signals comprises routing the control and data signals through a slave interface using muxes.

24. (Original) A method according to claim 21, wherein each of the DMA capable ASIC blocks comprises a master interface configured to initiate reads and writes to the system memory and a slave interface configured to read and write registers that setup and run the DMA.

25. (Original) A method according to claim 21, wherein the second DMA capable ASIC block uses a split response to pause the DMA in the initiating ASIC block until the second block is ready to begin the transfer.

26. (Original) A method according to claim 21, wherein the initiating DMA capable ASIC block is set up as if it is going to read or write to a specific address.

27. (Currently Amended) A method of transferring data from a first DMA capable ASIC block to a second DMA capable ASIC block across a bus, the method comprising:

mapping a slave interface of the first block to a master interface of the second block; and

initiating a transfer of the data at the first logic block while holding off the transfer of the data at the second block until the second block is set up and ready to complete the transfer,

wherein both the first block and the second block are set up as if they had been granted bus access even though only one block actually has access granted from a bus arbiter.

28. (Original) A method according to claim 27, wherein the transfer of data is held off by sending a split response until the second block is set up and ready to complete the transfer.

29. (Currently Amended) A method according to claim 27, wherein ~~both the first block and the second block are set up as if they had been granted bus access even though only one block actually has access granted from a bus arbiter~~ transfer of data between the first block and the second block takes place without accessing system memory.

30. (Currently Amended) A method according to claim ~~29~~ 27, wherein the first block has bus access granted by the bus arbiter and wherein the second block uses internal logic to cause it to operate as if it has bus access.